AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently amended) A system for interacting with an object, the system comprising:

a method call interceptor ~~operable to~~ that intercepts a method call to an object and [[to]] that routes the method call to a proxy, the method call interceptor accessible to application code; and

an application code generic proxy that ~~operable to~~ receives an intercepted method call, ~~the application code generic proxy~~ ~~further operable to~~ invokes a method on the object, [[to]] receives results from the object and [[to]] passes results to the entity that generated the intercepted method call based at least in part on the intercepted method call operability of the application code generic proxy modified by the application code.

2. (Currently amended) The system of claim 1, ~~where~~ the object is located across a remoting boundary.

3. (Currently amended) The system of claim 2, ~~where~~ the object is marshaled by reference.

4. (Currently amended) The system of claim 2, ~~where~~ the object is marshaled by value.

5. (Currently amended) The system of claim 1, ~~where~~ the method call interceptor ~~is~~ ~~further operable to~~ populates a call information data store with information associated with the intercepted method call, the call information data store is accessible to the application code generic proxy.

6. (Currently amended) The system of claim 5, ~~where~~ the call information data store is populated with at least one of a method name, one or more input parameters, a count of the number of input parameters, one or more type identifiers associated with the input parameters, a count of the number of return parameters for the method call, one or more type identifiers associated with the return parameters, class/interface defining method data, a stack pointer and a heap pointer.

7. (Currently amended) The system of claim 6, ~~where~~ the call information data store is a message object that can be serialized and passed across a remoting boundary.

8. (Currently amended) The system of claim 1, ~~where~~ the method call interceptor ~~is further operable to~~ transfers control to a method in the application code generic proxy, ~~where~~ the method in the application code generic proxy overrides a base class method defined in a base class object from which the application code generic proxy inherits.

9. (Currently amended) The system of claim 1, ~~where~~ the application code generic proxy ~~is operable to~~ performs proxy pre-processing before invoking the method on the object.

10. (Currently amended) The system of claim 9, ~~where~~ the proxy pre-processing comprises at least one of load-balancing, transaction processing, object migration, object persisting, monitoring remote method calls, caching local data, caching remote data, controlling remote method call invocations and machine learning involved in optimizing remote method call invocation.

11. (Currently amended) The system of claim 1, ~~where~~ the application code generic proxy ~~is operable to~~ performs proxy post-processing after receiving the results from the object.

12. (Currently amended) The system of claim 11, ~~where~~ the proxy post-processing comprises at least one of auditing, transaction processing, object migration, object persisting, monitoring remote method calls, caching local data, caching remote data, controlling remote method call invocations and machine learning involved in optimizing remote method call invocation.

13. (Currently amended) The system of claim 1, ~~where~~ the application code generic proxy invokes the method on the object by invoking a method available in a remoting infrastructure.

14. (Currently amended) A computer readable medium containing computer executable components ~~for a system~~ for interacting with an object, the components comprising:

a method call intercepting component that ~~operable to~~ intercepts a method call to an object and [[to]] routes the method call to a proxy, the method call intercepting component accessible to application code; and

an application code generic proxy component that ~~operable to~~ receives an intercepted method call, ~~the application code generic proxy component further operable to~~ invokes a method on the object, [[to]] receives results from the object and [[to]] passes results to the entity that generated the intercepted method call, the application code generic proxy component functionality modified by the application code based at least in part on the intercepted method call.

15. (Currently amended) A method for interacting with an object, the method comprising:

creating a base class proxy object;

creating an application code generic proxy, ~~where~~ the application code generic proxy inherits from the base class proxy object;

overriding a base class method defined in the base class, ~~where~~ the overridden method [[will]] receives an intercepted method call;

intercepting a method call on the object;

routing the method call to the application code generic proxy;

adapting the application code generic proxy functionality based at least in part on the method call;

invoking a method on the object;

receiving a first result from the object; and

returning a second result to the entity that generated the intercepted method call.

16. (Currently amended) The method of claim 15, ~~where~~ the application code generic proxy performs proxy pre-processing before invoking the method on the object.

17. (Currently amended) The method of claim 16, ~~where~~ the proxy pre-processing comprises at least one of load-balancing, transaction processing, object migration, object persisting, monitoring remote method calls, caching local data, caching remote data, controlling remote method call invocations and machine learning involved in optimizing remote method call invocation.

18. (Currently amended) The method of claim 15, ~~where~~ the application code generic proxy performs proxy post-processing before returning the result to the entity that generated the intercepted method call.

19. (Currently amended) The method of claim 18, ~~where~~ the proxy post-processing comprises at least one of auditing, transaction processing, object migration, object persisting, monitoring remote method calls, caching local data, caching remote data, controlling remote method call invocations and machine learning involved in optimizing remote method call invocation.

20. (Currently amended) The method of claim 15, ~~where~~ the object is located across a remoting boundary.

21. (Currently amended) The method of claim 20, ~~where~~ the object is marshaled by reference.

22. (Currently amended) The method of claim 20, ~~where~~ the object is marshaled by value.

23. (Currently amended) A computer readable medium containing computer executable instructions for performing a method for interacting with an object, the method comprising:

creating a base class proxy object;

creating an application code generic proxy, ~~where~~ the application code generic proxy inherits from the base class proxy object;

overriding a base class method defined in the base class, ~~where~~ the overridden method [[will]] receives an intercepted method call;

intercepting a method call on the object;

based at least in part on the intercepted method call, adjusting the application code generic proxy functionality;

routing the method call to the application code generic proxy;

invoking a method on the object;

receiving a first result from the object; and

returning a second result to the entity that generated the intercepted method call.

24. (Previously presented) A data packet adapted to be transmitted between two or more computer processes, the data packet comprising:

one or more identifier/value pairs, the identifier identifying the value associated with the identifier/value pair, and the value providing information associated with an intercepted method call on an object;

wherein the information associated with an intercepted method call on an object comprises at least one of method name, one or more input parameters, a count of the number of parameters input to the method, one or more type identifiers associated with the input parameters, a count of the number of return parameters for the method, one or more type identifiers associated with the return parameters, class/interface defining method data, a stack pointer and a heap pointer.

25. (Cancelled).

26. (Currently amended) The data packet of claim 24, ~~where~~ the data packet is a serializable message object that can be passed across a remoting boundary.

27. (Currently amended) A system for interacting with an object, the system comprising:

means for creating a base class proxy object from application code, ~~where~~ the base class proxy object has a method that can be overridden by an inheriting application code generic proxy so that the overridden method can receive an intercepted method call;

means for creating the application code generic proxy, ~~where~~ the application code generic proxy inherits from the base class proxy object and ~~where~~ the application code generic proxy overrides the base class method that can be overridden;

means for intercepting a method call and for transferring control to the overridden base class method in the application code generic proxy;

means for the application code generic proxy to receive the intercepted method call;

means for modifying the application code generic proxy operability based at least in part on the intercepted method call and externally supplied application code;

means for providing the overridden base class method with a call data structure associated with the intercepted method call;

means for the application code generic proxy to invoke the method on the object;

means for the application code generic proxy to receive a first result from the object; and

means for the application code generic proxy to return a second result to the entity that generated the intercepted method call.